UNIDAD 4: ESTRUCTURAS.

4.1 *ESTRUCTURA SELECTIVA*

De acuerdo con las condiciones de un problema en ocasiones se necesita tener la posibilidad de controlar cuales son las instrucciones que se deben ejecutar y en qué momento, para esto los compiladores traen instrucciones que le permiten al programador elegir entre una o varias alternativas.

Las condicionales son expresiones booleanas, que se emplean para controlar la secuencia en que deben ejecutarse los pasos de un algoritmo. Generan un valor verdadero (1) o falso (0), esto es pueden tomar dos valores: verdadero (true) y falso (FALSE).

Las instrucciones condicionales controlan si una instrucción o lista de instrucciones se ejecutan en función del cumplimiento o no, de una condición

Esta instrucción la denominaremos **si** y tiene la siguiente forma:

```
si (condición)
        entonces instrucciones
        sino  instrucciones
fin_si
```

La condición es una expresión booleana, si el valor generado es verdadero entonces se ejecutan las instrucciones que siguen a la palabra entonces. Si el valor generado por la condición es falso, se ejecutan las instrucciones que siguen a la palabra sino, en caso de que no se requieran especificar instrucciones por falso, simplemente se omite esta parte y las instrucciones por verdadero no se ejecutarían:

```
si (condición)
  entonces instrucciones
fin_si
```

EJEMPLO 1.

```
Inicio

    Leer x, y
        si (x<y)
            entonces x ← x+1
        fin_si
    Escribir x,y
 Fin
```

En el ejemplo anterior la expresión booleana es (x<y), si el valor generado es verdadero (Por ejemplo si $x=2$ y $y=3$) entonces se ejecuta la instrucción a x asígnele $x+1$; en caso

de que la expresión booleana sea falsa (por ejemplo *x*=3 y *y*=2) entonces la secuencia de instrucción a ejecutar seria la siguiente a fin_si, esto es escribir *x,y*.

EJEMPLO 2.

```
Inicio

  Leer x, y
      si (x<y)
        entonces x ← x+1
        sino  y ← y+1
      fin_si
  Escribir x,y
 Fin
```

Este ejemplo se diferencia del anterior en que cuando la condición es falsa se debe realizar a *y* asígnele *y*+1 y luego sin importar que la condición fuera verdadera o falsa se escribe *x,y*.

CLASES DE CONDICIONES

Existen dos clases de condiciones, las condicionales simples y las condiciones compuestas.

– **CONDICIONES SIMPLES**

Son condiciones sencillas que establecen una relación entre dos constantes, dos variables, o una variable con una constante, utilizando los operadores de relación. Por ejemplo:

A >= B
X != 0

– **CONDICIONES COMPUESTAS**

Permiten enlazar condiciones simples para formar otras más complejas; las condiciones son enlazadas entre sí por medio de los operadores lógicos. Por ejemplo:

a > b y a > c
x != y ó x = z
no (a > b) y no (x = y)

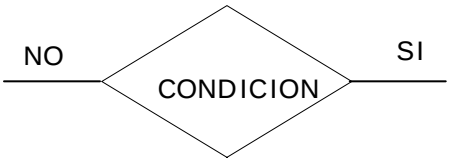
EJEMPLO:

```
Inicio
 Leer a
 si (a<10 o a>20)
       entonces x ← 3
       sino  x ← 0
 fin_si
 Escribir x
Fin
```

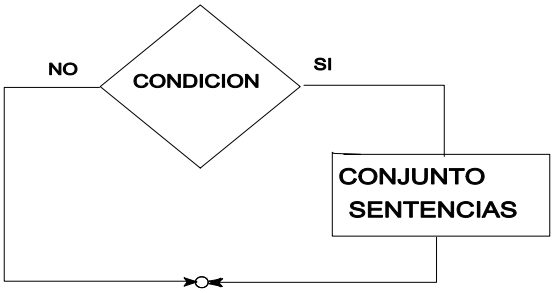
En el ejemplo anterior si la condición es verdadera (por ejemplo si a=3) se le asigna 3 a *x*; en caso de que sea falso (por ejemplo *x*=15) se le asigna 0 a *x*.

Representación de los condicionales en diagramas de flujo

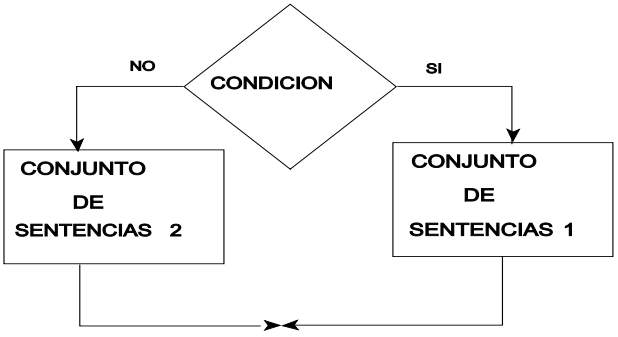
Para representar un condicional se utiliza un rombo:



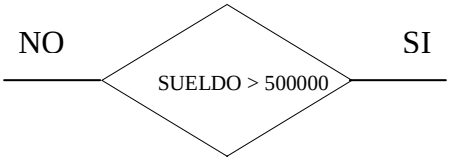
En caso que no se quieran especificar instrucciones por falso:



Cuando se quieran especificar instrucciones por falso y verdadero:

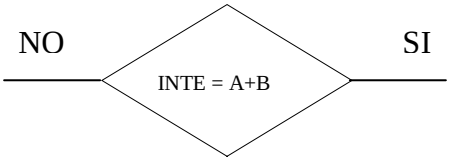


A continuación presentamos unos ejemplos gráficos de enunciados de decisión:

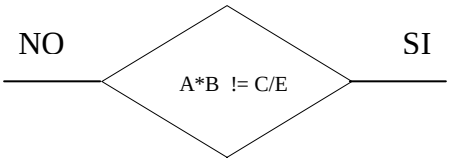
COMPOSICION	**EJEMPLO**
Variable: Constante	

Variable: Variable

Variable: Expresión



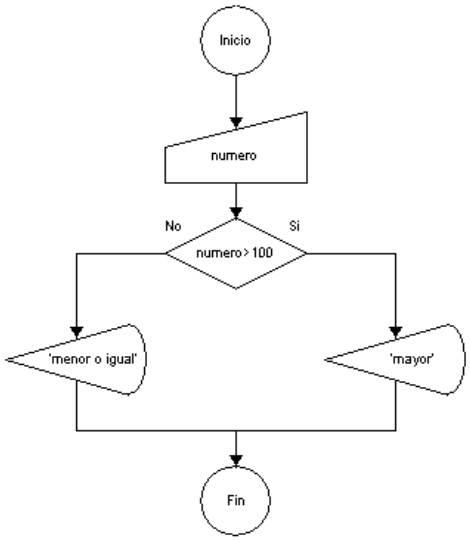
Expresión: Expresión



EJEMPLO 1.

Leer un número y escribir si es "mayor" o "menor o igual" que 100

PSEUDOCODIGO	DIAGRAMA DE FLUJO

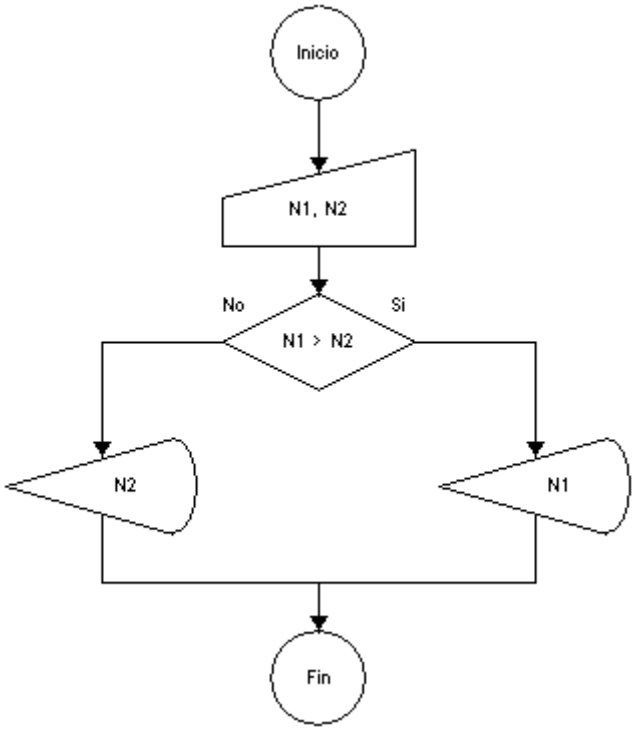
Inicio
  Leer numero
  si (numero > 100)
  entonces escribir "mayor"
  sino escribir "menor o igual"
  fin_si
Fin
	

EJEMPLO 2.

Leer dos números, y determinar cual es el mayor.

Variables: N1: Numero
N2: Numero

PSEUDOCODIGO	DIAGRAMA DE FLUJO

Inicio
  Leer N1, N2
  si (N1 > N2)
  entonces escribir N1
  sino escribir N2
  fin_si
Fin
	

CONDICIONALES ANIDADOS

En ocasiones es necesita tener la posibilidad de controlar las instrucciones que se deben ejecutar entre más de una alternativa, en cumplimiento de dos o más condiciones. La instrucción **si**, puede incluir otras instrucciones **si**. En este caso se dice que las instrucciones **si** están anidadas.

Esta instrucción **si anidada**, tiene la siguiente forma:

si (condición1)
  entonces instrucciones
  sino
    si (condición2)
      entonces instrucciones
    sino instrucciones
    fin_si
fin_si

En la forma anterior, se tienen dos condiciones para tres posibilidades de instrucciones.

Nótese que en la negación de la primera condición, resulta el segundo **si**, que nos permite evaluar las dos posibilidades de la segunda condición.

Otra forma sería que el segundo **si**, estuviera por la afirmación de la primera condición, con lo cual tendría la siguiente apariencia:

```
si (condición1)
    entonces  si (condición2)
                    entonces instrucciones
                    sino  instrucciones
                fin_si
    sino instrucciones
fin_si
```

Se acepta un número indeterminado de **si** anidados, que depende de la habilidad del programador para controlar todas las posibilidades. A continuación se presenta un ejemplo de cuatro condicionales:

```
si (condición1)
    entonces si (condición2)
                entonces si (condición3)
                        entonces  si (condición4)

                                entonces instrucciones1
                            sino  instrucciones2
                            fin_si
                        sino instrucciones3
                        fin_si
                sino instrucciones4
            fin_si
    sino instrucciones5
fin_si
```

EJEMPLO 1:

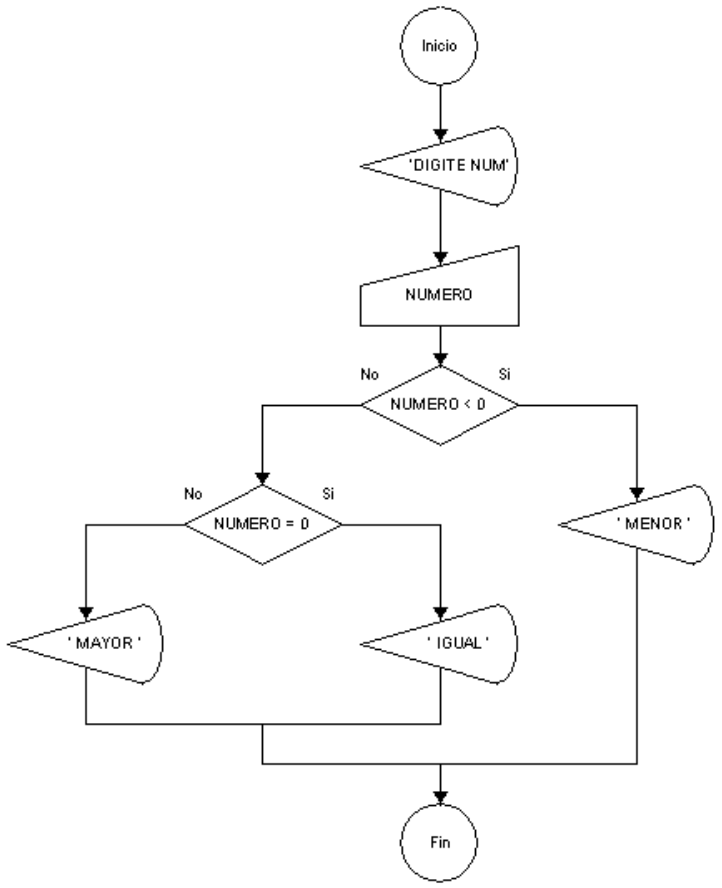
Leer un número y escribir si es menor que cero, igual a cero o mayor que cero.

Variable: NUMERO

PSEUDOCODIGO

```
Inicio
  leer NUMERO
  si (NUMERO < 0)
      entonces escribir "menor que cero"
      sino  si (NUMERO = 0)
                entonces    escribir "igual a cero"
                sino escribir "mayor que cero"
            fin_si
  fin_si
fin
```

DIAGRAMA DE FLUJO



EJEMPLO 2:

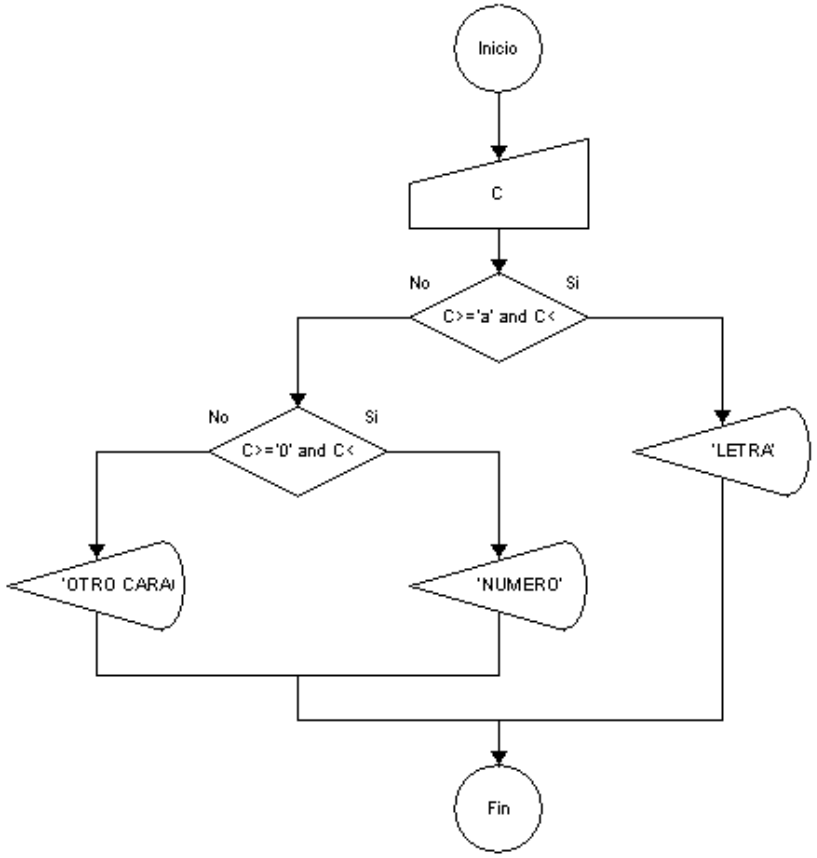
Leer un carácter, determinar si es una letra, un número, otro carácter.

Variables: C: Carácter

PSEUDOCODIGO

```
Inicio
  leer C
  si (C>='a' y C<='z')
      entonces  escribir "LETRA"
      sino si (C>='0' y C<='9')
                entonces escribir "NUMERO"
                sino escribir "OTRO CARACTER"
            fin_si
  fin_si
fin
```

DIAGRAMA DE FLUJO



EJEMPLO 3:

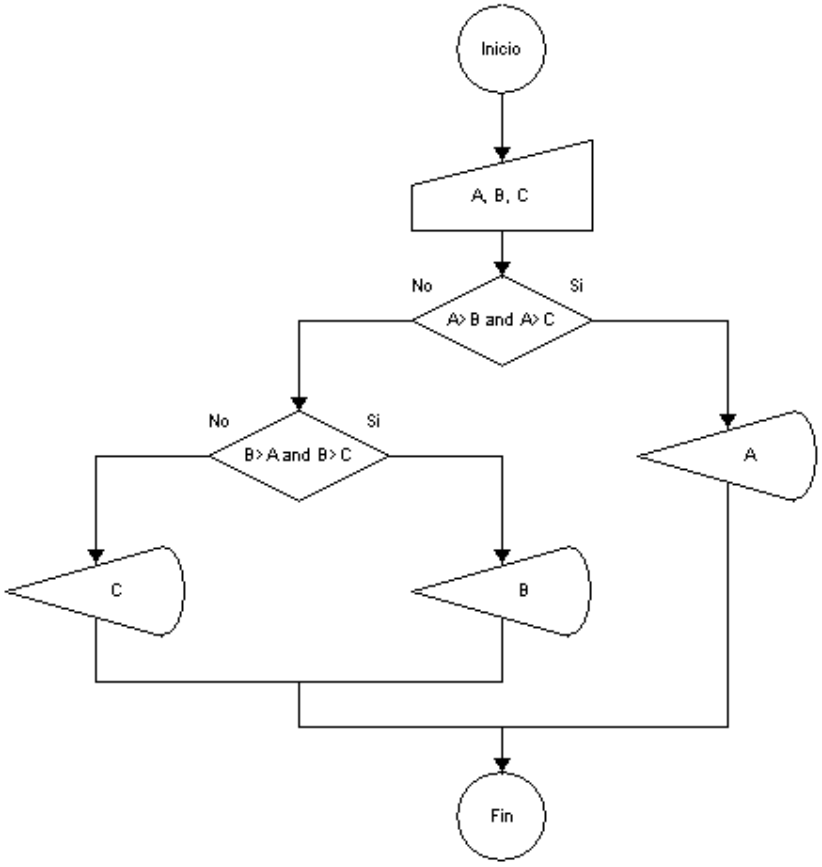
Leer tres números, determinar cual es el mayor.

Variables: A: Numero 1, B: Numero 2, C: Numero 3

PSEUDOCODIGO

```
Inicio
  leer A, B, C
  si (A > B y A>C)
      entonces escribir A
      sino
      si (B>A y B>C)
         entonces escribir  B
         sino
            escribir C
      fin_si
  fin_si
fin
```

DIAGRAMA DE FLUJO



4.2 *ESTRUCTURAS REPETITIVAS*

En la elaboración de programas es importante que existan instrucciones que permiten que el computador ejecute un grupo de instrucciones varias veces sin tener que volverlas a escribir, estas son: la instrucción **mientras que** y la instrucción **para**.

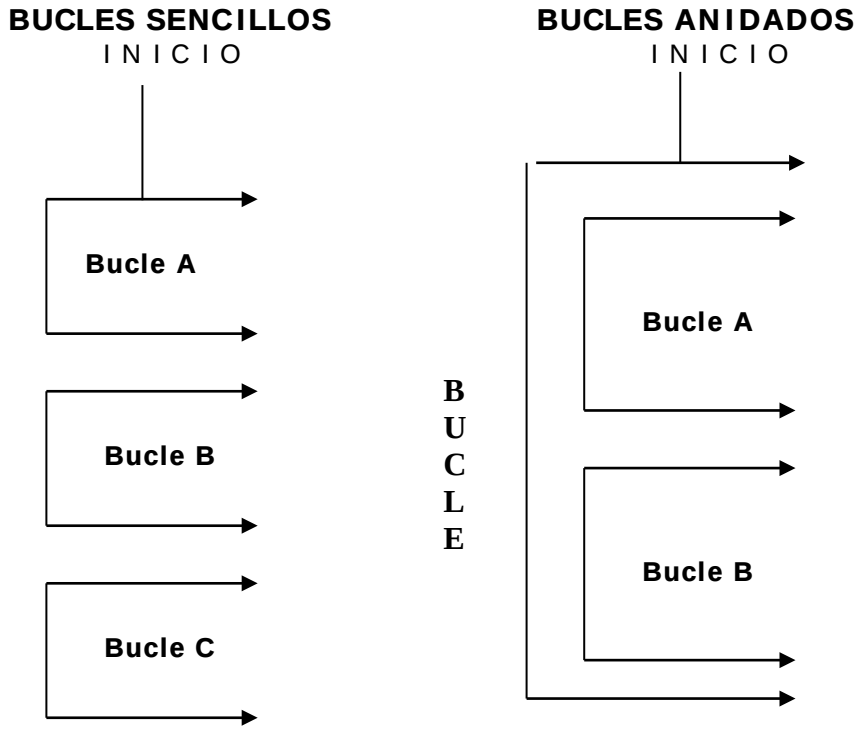
4.2.1 BUCLES E ITERACIONES

Un bucle o ciclo es un pedazo de algoritmo, cuyas instrucciones se repiten un número determinado de veces mientras se cumple una determinada condición.

Un bucle consta de tres partes fundamentales: DECISION (simple o compuesta), CUERPO DEL BUCLE y la SALIDA DEL BUCLE.

Una iteración es la ejecución de todas las instrucciones del ciclo.

Los bucles pueden ser sencillos o anidados (ciclos dentro de otros ciclos).



CONTADOR

Los procesos repetitivos son la base del uso del computador. En estos procesos se necesita normalmente contar los sucesos o acciones internas del bucle. Una forma de controlar un bucle es mediante un contador.
Un contador es una variable cuyo valor se incrementa o decrementa en una cantidad constante por cada iteración.

El formato general es: CONTADOR ← CONTADOR + CANTIDAD CONSTANTE

Ejemplo:

Z ← Z + 2, donde Z se incrementa en 2 en cada iteración.

Acumulador

Es una variable cuya misión es almacenar por cada iteración, cantidades variables resultantes de sumas o productos sucesivos.

El formato general es: ACUMULADOR ← ACUMULADOR + CANTIDAD VARIABLE ó
ACUMULADOR ← ACUMULADOR * CANTIDAD VARIABLE

Ejemplo:

S ← S+N, donde N es una variable y no una constante.

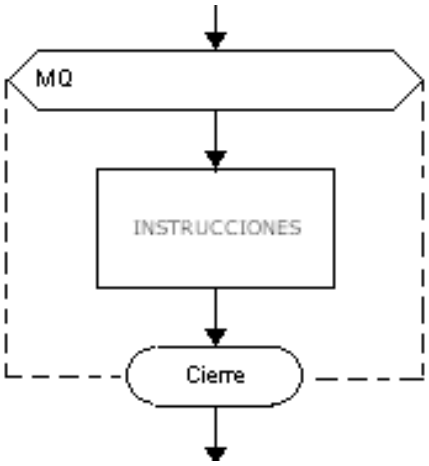
4.2.2 *INSTRUCCIÓN MIENTRAS QUE*

La instrucción **mientras**, nos permite ejecutar una instrucción o un grupo de instrucciones un número determinado de veces, mientras se cumple una condición. Esta instrucción es utilizada, cuando no se sabe el número de veces que va a ocurrir el ciclo.

La representamos en pseudocódigo de la siguiente forma:

mientras (condición) **hacer**
    instrucciones
fin_mientras

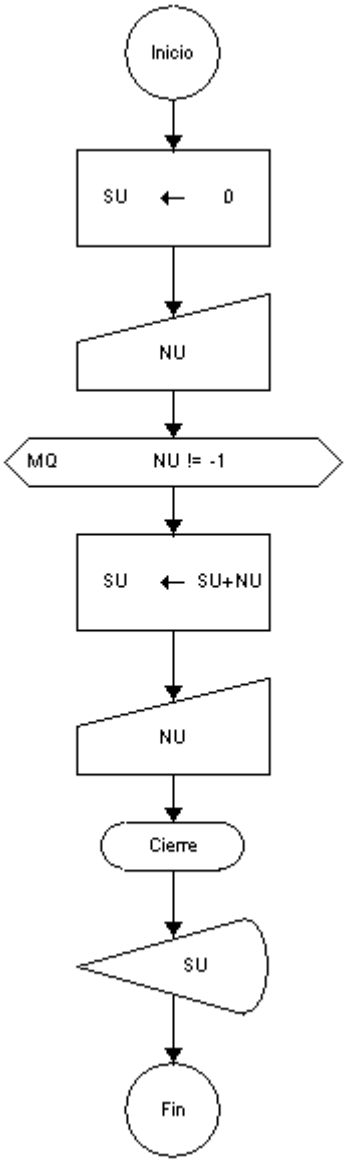
En los diagramas de flujo la representamos con un rectángulo terminado en puntas:



EJEMPLO 1.

Sumar una serie de números, el programa solicita los números por teclado hasta que se digite el numero -1

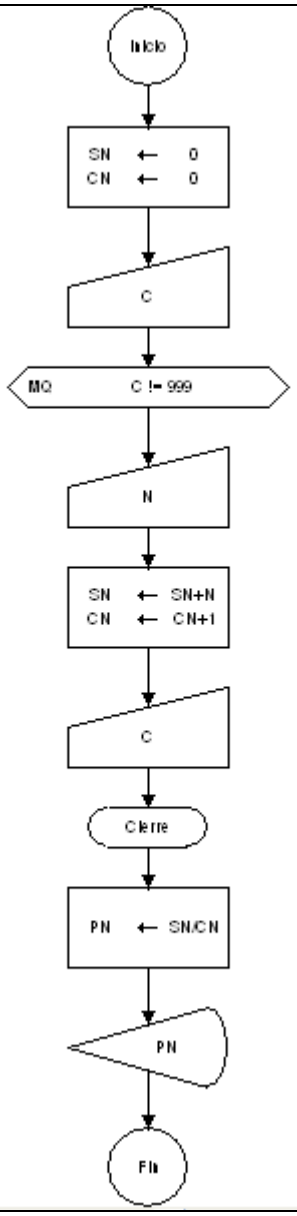
Variables: SU: Acumulador (suma de los números), NU: Numero

PSEUDOCODIGO	DIAGRAMA DE FLUJO
Inicio SU ← 0 **Leer** NU **mientras** (NU != -1) **hacer** SU ← SU + NU **leer** NU **fin_mientras** **Escribir** SU **Fin**	

EJEMPLO 2.

Calcular el promedio de las notas de un grupo de alumnos. El programa finaliza el ingreso de notas cuando se digite el código 999.

Variables: N: Nota, SN: Suma de notas, C: Código, CN: Cantidad de notas, PN: Promedio de notas

PSEUDOCODIGO	DIAGRAMA DE FLUJO
Inicio SN ← 0 CN ← 0 **Leer** C **mientras** (C != 999) **hacer** **leer** N SN ← SN + N CN ← CN + 1 **leer** C **fin_mientras** PN ← SN / CN **Escribir** PN **Fin**	

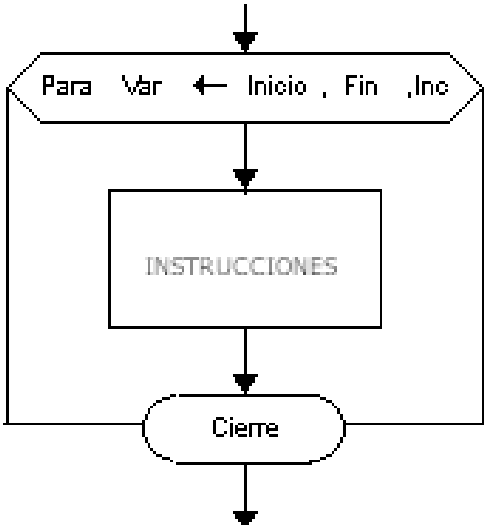
4.2.3 INSTRUCCIÓN PARA

La instrucción **para** permite repetir una instrucción o un grupo de instrucciones un número determinado de veces. Esta instrucción solo es utilizada cuando se sabe el número exacto de veces que va a ocurrir un ciclo.

La representamos en el pseudocódigo de la siguiente forma:

para Variable ← valor_inicial , valor_final , incremento **haga**
instrucciones
fin_para

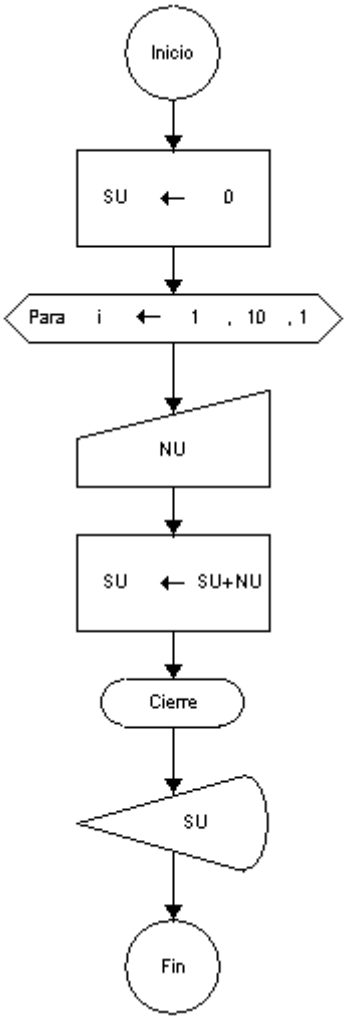
En los diagramas de flujo la representamos con un rectángulo terminado en puntas:



Ejemplo 1.

Sumar 10 números introducidos por teclado:

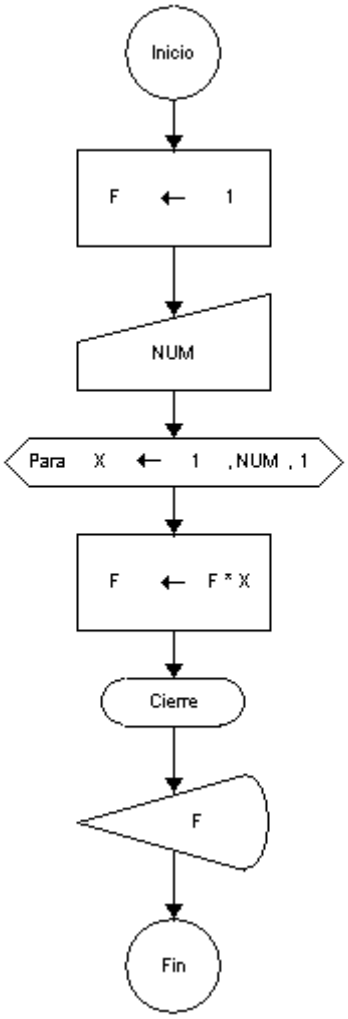
Variables: SU: Acumulador (Suma de los números), NU: Numero que es introducido por teclado, i: Contador que controla el ciclo para

PSEUDOCODIGO	DIAGRAMA DE FLUJO
Inicio SU ← 0 **para** i ← 1, 10, 1 **haga** **Leer** NU SU ← SU + NU **fin_para** **Escribir** SU **Fin**	

Ejemplo 2.

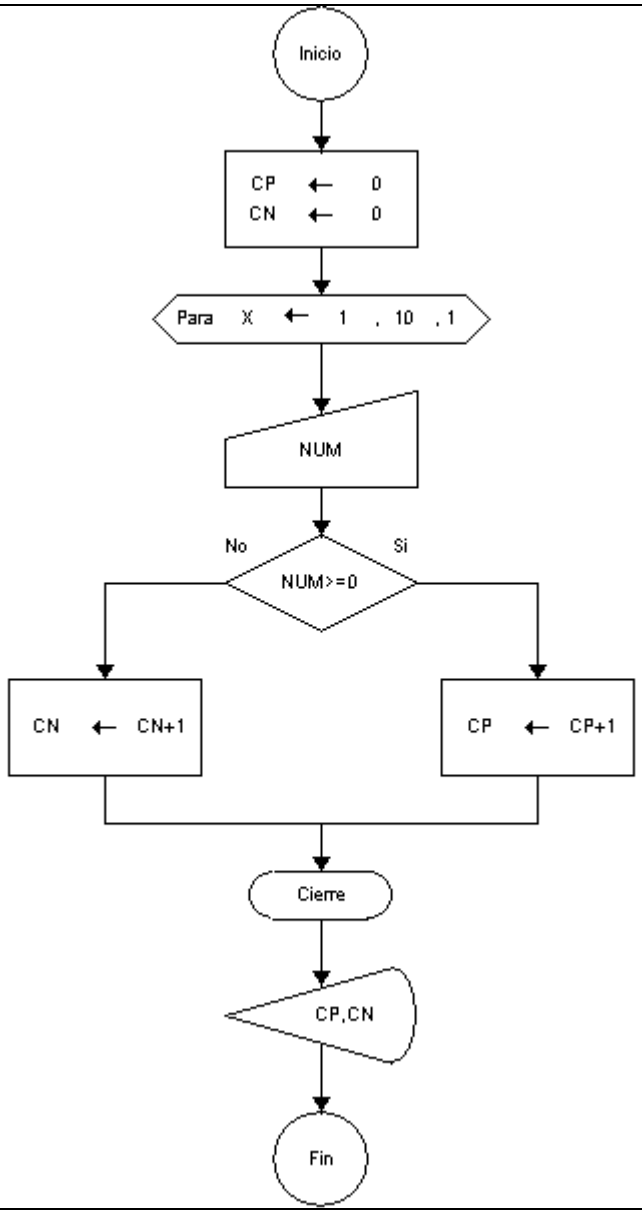
Calcular y escribir el factorial de un número dado por el usuario.

Variables: NUM: Numero, F: Factorial

PSEUDOCODIGO	DIAGRAMA DE FLUJO
Inicio F ← 1 **Leer** NUM **para** X ← 1, NUM, 1 **haga** F ← F * X **fin_para** **escribir** F **Fin**	

EJEMPLO 3

Leer 10 números, calcular y escribir la cantidad de números positivos y la cantidad de números negativos.

Variables: NUM: Numero, CP: Cantidad de números positivos, CN: Cantidad de números negativos.

PSEUDOCODIGO	DIAGRAMA DE FLUJO
Inicio CP ← 0 CN ← 0 **para** X ← 1, 10, 1 **haga** **Leer** NUM **si** (NUM>=0) **Entonces** CP ← CP + 1 **sino** CN ← CN + 1 **fin_si** **fin_para** **Escribir** CP, CN **Fin**